

AMENDMENTS TO THE CLAIMS:

This listing of claims will replace all prior versions and listings of claims in the application:

1. (Currently Amended) A method for validating programs, the method comprising:

receiving a language-independent description of a computer program, the language-independent description comprising a definition module and an implementation module, the implementation module defining a class to be implemented by the program and the definition module defining an interface associated with the class;

validating the language-independent description;

generating a language-dependent program from the language-independent description, the language-dependent program comprising ~~an~~ the interface and a the class; and

validating the language-dependent program.

2. (Original) The method of claim 1 wherein validating the language-independent description comprises validating the syntax of the definition module and the implementation module.

3. (Original) The method of claim 1 wherein validating the language-dependent program comprises compiling the interface and the class.

4. (Original) The method of claim 1 wherein the definition module and the implementation module are represented in a meta-language or using a tree structure.

5. (Currently Amended) A method for validating programs, the method comprising:

receiving a language-independent description of a computer program, the language-independent description comprising a definition module and an implementation module;

validating the language-independent description;

generating a language-dependent program from the language-independent description, the language-dependent program comprising a script code section in a language that does not support interfaces; and

validating the language-dependent program.

6. (Original) The method of claim 5 wherein validating the language-dependent program comprises:

extracting language elements from the script code section; and

comparing the extracted language elements with the definition module.

7. (Original) The method of claim 6 wherein extracting language elements comprises generating a symbol table from the script code section.

8. (Original) The method of claim 5 wherein generating the language-dependent program comprises:

generating language-dependent code comprising an interface and a class.

9. (Original) The method of claim 5, wherein validating the language-dependent program comprises:

extracting language elements from the script code section;

comparing the extracted language elements with the definition module;

generating language-dependent code comprising an interface and a class; and

compiling the interface and the class.

10. (Currently Amended) A method for validating programs, the method comprising:

receiving a language-independent description of a computer program, the language-independent description comprising a definition module and an implementation module;

validating the language-independent description;

generating a first language-dependent program from the language-independent description, the first language-dependent program comprising a first script code section in a language that does not support interfaces;

generating a second language-dependent program from the ~~language-dependent~~ language-independent description, the second language-dependent program comprising a second script code section of a distinct, second kind in a language that does not support interfaces;

extracting a first set of language elements from the first script code section;

extracting a second set of language elements from the second script code section; and

comparing the first set of language elements and the second set of language elements with the definition module.

11. (Currently Amended) A computer program product, tangibly embodied in an ~~information carrier~~ a computer-readable storage device, the computer program product comprising instructions operable to cause data processing equipment to:

receive a language-independent description of a computer program, the language-independent description comprising a definition module and an implementation module, the implementation module defining a class to be implemented by the program and the definition module defining an interface associated with the class;

validate the language-independent description;

generate a language-dependent program from the language-independent description, the language-dependent program comprising an the interface and a the class; and

validate the language-dependent program.

12. (Original) The computer program product of claim 11, wherein the instructions to validate the language-independent description cause the data processing equipment to validate the syntax of the definition module and the implementation module.

13. (Original) The computer program product of claim 11, wherein the instructions to validate the language-dependent program cause the data processing equipment to compile the interface and the class.

14. (Original) The computer program product of claim 11 wherein the definition module and the implementation module are represented in a meta-language.

15. (Currently Amended) A computer program product, tangibly embodied in ~~an information carrier~~ a computer-readable storage device, the computer program product comprising instructions operable to cause data processing equipment to:

receive a language-independent description of a computer program, the language-independent description comprising a definition module and an implementation module;

validate the language-independent description;

generate a language-dependent program from the language-independent description, the language-dependent program comprising a script code section in a language that does not support interfaces; and

validate the language-dependent program.

16. (Original) The computer program product of claim 15, wherein the instructions to validate the language-dependent program cause the data processing equipment to:

extract language elements from the script code section; and

compare the extracted language elements with the definition module.

17. (Original) The computer program product of claim 16 wherein the instructions to extract the language elements cause the data processing equipment to generate a symbol table from the script code section.

18. (Original) The computer program product of claim 15, wherein the instructions to generate the language-dependent program cause the data processing equipment to:

generate language-dependent code comprising an interface and a class.

19. (Original) The computer program product of claim 15 wherein the instructions to validate the language-dependent program cause the data processing equipment to:

extract language elements from the script code section;

compare the extracted language elements with the definition module;

generate language-dependent code comprising an interface and a class; and

compile the interface and the class.

20. (Currently Amended) A computer program product, tangibly embodied in ~~an information carrier~~ a computer-readable storage device, the computer program product comprising instructions operable to cause data processing equipment to:

receive a language-independent description of a computer program, the language-independent description comprising a definition module and an implementation module;

validate the language-independent description;

generate a first language-dependent program from the language-independent description, the first language-dependent program comprising a first script code section in a language that does not support interfaces;

generate a second language-dependent program from the ~~language-dependent~~ language-independent, the second language-dependent program comprising a second

script code section of a distinct, second kind in a language that does not support interfaces;

extract a first set of language elements from the first script code section;

extract a second set of language elements from the second script code section;

and

compare the first set of language elements and the second set of language elements with the definition module.

21. (Currently Amended) An apparatus, comprising:

means for receiving a language-independent description of a computer program, the language-independent description comprising a definition module and an implementation module, the implementation module defining a class to be implemented by the program and the definition module defining an interface associated with the class;

means for validating the language-independent description;

means for generating a language-dependent program from the language-independent description, the language-dependent program comprising an the interface and a the class; and

means for validating the language-dependent program.